Formal Analysis of Sensor Network Encryption Protocol (SNEP)

Llanos Tobarra, Diego Cazorla and Fernando Cuartero
Real Time And Concurrent System Group (ReTICS)
University of Castilla-La Mancha
Escuela Politécnica Superior de Albacete
Albacete, Spain – 02071
Email: {mtobarra, dcazorla, fernando}dsi.uclm.es

***Abstract*—In this paper, a formal analysis of a security protocol in the field of wireless sensor networks is presented. Sensor Network Encryption Protocol (SNEP) describes basic primitives for providing confidentiality, authentication between two nodes, data integrity and weak message freshness in a wireless sensor network. It was designed as base component of Security Protocols for Sensor Networks (SPINS). SNEP is modelled in two scenarios using the high-level formal language HLPSL, and verified using the model checking tool Avispa, where two main security properties are checked: authenticity and confidentiality of relevant messages components. The first case is the communication between the base station and networks nodes in order to retrieve node confidential information. The second case is a key distribution protocol in a sensor network using SNEP for securing messages. As a result of this analysis, one attack have been found: a false request message from an intruder. In that case, the intruder impersonates the base station and creates false requests. This way, the intruder may obtain confidential data from a node in the network. A solution to this attack is proposed in the paper.**

***Index Terms*—Wireless sensor, model checking, security protocols, avispa toolbox, SNEP**

I. Introduction

Security has become a challenge in wireless sensor networks. Low capabilities of devices, in terms of computational power and energy consumption, make difficult using traditional security protocols.

Two main problems related to security protocols arise. Firstly, the overload that security protocols introduce in messages should be reduced at a minimum; every bit the sensor sends consumes energy and, consequently, reduces the life of the device. Secondly, low computational power implies that special cryptographic algorithms that require less powerful processors need to be used. The combination of both problems lead us to a situation where new approaches or solutions to security protocols need to be considered. These new approaches take into account basically two main goals: reduce the overhead that protocol imposes to messages, and provide reasonable protection while limiting use of resources.

This work has been supported by the spanish government with the project ”Application of Formal Methods to Web Services”, with reference TIN2006-15578-C02-02, and the JCCM regional project “Application of formal methods to the design and analysis of Web Services and e-commerce ” (PAC06-0008-6995)

In order to design a secure network, several aspects have to be considered [1]: Key establishment and trust setup, secrecy and authentication, and privacy. Key establishment can be considered the base of the system; a secure and efficient key distribution mechanism is needed for large scale sensor networks. Once every node has its own keys, these are used to authenticate and encrypt (if needed) the messages they exchange. Several protocols have been proposed in the literature related to authentication and privacy [2], [3], and key distribution [4], [5], [6].

In this paper, we have focus in one of these protocols: SNEP [7]. A brief overview of this protocol is given in Section III. SNEP describes basic primitives for providing confidentiality, authentication between two nodes, data integrity and weak message freshness in a wireless sensor network.

The paper is organised as follows. In Section II an introduction of formal methods for the analysis of security protocols is presented. In Section III a brief overview of SNEP is given. Section IV is devoted to the formal verification of SNEP, where two scenarios have been considered depending on the key distribution mechanism used. Finally in Section V we give our conclusions and future work.

II. Model checking for the analysis of security protocols

As new security protocols are designed, new techniques have also been developed to model a system and check properties on it. One of the most promising techniques in this line is *model checking*. Model checking [8] is a technique based on formal methods for verifying finite-state-concurrent systems, and has been implemented in several tools. One of the main advantages of this technique is that it is automatic and allows us to see if a system works as expected. In case the system does not work properly, the model checking tool provides a trace that leads to the source of the error.

Model checking has become a key point in the design of concurrent and distributed system because it allows us to ensure the correctness of a design at the earliest stage possible. Model checking has two main advantages over two classical techniques such as simulation and testing: *i)* we do not need to build a prototype of the system, and *ii)* we are able to verify the system against every single execution trace. The latter is very important because using simulation or testing we can only find errors, but we cannot ensure that the whole system behaves as expected (some errors may remain hidden until the system is in production stage).

Some general purpose model checking tools have been developed by different research groups: Spin, UPPAAL, Murϕ, FRD2.0, etc. These tools allow us to verify not only the functional properties of a system (e.g. Spin), but also the performance of a real-time system (e.g. UPPAAL). Although we can use these general purpose tools in order to verify security protocols, we consider that it is preferable (and more intuitive) to use a tool devoted to the verification of security protocols. Among these tools we can find Casper/FDR2 toolbox [9] and AVISPA [10].

The use of model checking tools to verify security protocols has been successful in the past in different areas such as Web Services [11], [12], [13], wireless security protocols [14], or Transport Layer security protocols [15], [16].

a) AVISPA toolbox: As it is mentioned in the previous section, AVISPA provides a high-level formal language HLPSL [17] for specifying protocols and their security properties. Once we have specified the model of the system, AVISPA translates it into an intermediate format IF. This is the input of several backends that are integrated into AVISPA framework: SATMC, OFMC, Cl-Atse and TA4SP. Besides, only one model is specified although it can be analysed with the four backends. AVISPA also offers a graphical interface SPAN [18] that helps in the specifying task.

In AVISPA we find two kind of roles; basic roles and composed roles. A *basic role* describes a protocol participant initial knowledge, its initial state and a set of transitions that describes the behaviour of the participant. Transitions are a more expresive notation that A-B notation. It allows to express control flow elements as if-then-else structures, loops, etc. The basic roles can be composed in order to describe protocol sessions, which are composed roles.

Security in wireless networks is not an easy task due to its broadcast nature. An intruder can overhear, intercept messages, inject new messages or modify messages in transit. This kind of intruder is called Dolev-Yao Intruder [19]. The intruder implemented in AVISPA is a Dolev-Yao intruder, which is appropriate to analyse wireless security protocols.

b) Notation: We are going to use the following notation to describe the protocols and the models:

A, B, N_i	communicating nodes
BS	The base station
SK_{ab}	Symmetric key shared between nodes A and B
K_m	Master key shared between the node and the base station
K_{enc}	Encryption key derived from the master key
K_{mac}	MAC computing key derived from the master key
C_{a-b}	Shared counter between A and B
N_A	Nonce generated by A
$MAC(M)$	Message Authentication Code function computed over message M
$F(M)$	One-way hash function computed over message M
$\{M\}_K$	Message M encrypted with key K
$M_1 \oplus M_2$	XOR of messages M_1 and M_2
$Chan!M$	Sending message M on channel $Chan$
$Chan?M$	Receiving message M on channel $Chan$

III. Sensor Network Encryption Protocol: SNEP

SNEP is one of the two subprotocols that composed Security Protocols for Sensor Networks (SPINS). SPINS was designed for key distribution in sensor networks using SNEP and μTESLA as basic block component. In particular, SNEP describes basic primitives for providing confidentiality, authentication between two nodes, data integrity and weak message freshness. μTESLA is an adaptation of TESLA protocol for sensor networks. In this paper, we only focus on SNEP.

To fulfill semantic confidentiality [1], instead of initial vectors, SNEP uses shared counters. Every plain text block is ciphered with a counter usign count mode (CTR) encryption algorithm. The receiver and sender updates the shared counter once they just have received/sent a cipher block. Thus, it is not necessary to include the counter in the message. These shared counters offer a particial message order and a weak message freshness.

Each message includes a *Message Authentication Code* (MAC). It is computed with CBC-MAC algorithm over the ciphered data. It is computed once for each package. When an agent receives a message, it computes the message MAC and compares with the received MAC. If they are equal, the message is accepted. The MAC allows endpoints to prevent modifications of the message in transit. It also allows them to authenticate data origin because it is ciphered with a shared symmetric key between sender and receiver.

A message M between two nodes S and R secured with SNEP is the following:

$$S \rightarrow R : \{Ctr\}_{K_{enc}} \oplus M, MAC(K_{mac}, \{\{Ctr\}_{K_{enc}} \oplus M\})$$

K_{enc} and K_{mac} are derived keys from a master key K_m. This master key is shared with the base station and it is stored in the node before its deployment. The rest of keys are derived from the master key by means of a pseudo-random function

[1]Semantic security means that if the same plain-text is ciphered twice, the two resulting ciphetexts are different

$f_k(x) = MAC(k, x)$. If two nodes detect that the keys are compromised, they must generate a new pair.

IV. Verification of SNEP

We have considered two different configurations of SNEP on the same network configuration (see fig. 1), depending on the kind and the situation of the nodes that communicate with each other:

- *Case BS-N*: Request from the base node to a normal node.
- *Case N-N*: Key distribution between two nodes.

As is evident in fig. 1, the base station is the main gateway for nodes to communicate with outside world. Consequently, comprimising the base station means comprimising the whole network. Thus, the base station is a trusted component in the network. It is assumed it can not be impersonate. It is also assumed that the network is in a stable phase. The nodes do not dynamically connect to the network or they dynamically disconnect from the network. This assumption simplifies the analysis and it abstracts away the routing problems.

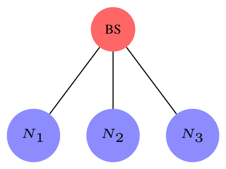

Figure 1. Sensor network for cases A and B

A. Case Base Station → Node

In this case, we analyse a request from the base station to the nodes. The nodes response to this request with relevant information.

As we mentioned in a previous section, SNEP only guarantees weak freshness. But, with this kind of freshness, there is not way to determine if a received message is generated as response of a previous event. Thus, SPINS propose the following protocol in order to fulfill strong freshness:

1. BS → N$_i$: $BS.N_i.Nonce.Msg_1$
2. N$_i$ → BS : $\{Ctr\}_{K_{enc}} \oplus Msg_2$
   $\{MAC(\{Ctr\}_{K_{enc}} \oplus Msg_2.Nonce.N_i.BS)\}_{K_{mac}}$

Where $Kenc = F(Km, 0)$ and $Kmac = F(Km, 1)$. Each node has a different value for K_m. The base estation BS has a list of pairs node N_i and a master key K_m. The base estation selects a node from its node list and sends an information request. In this information request it includes a nonce $Nonce$ as challenge and proof of freshness for the intended node. This nonce is included in the response MAC code. The fig. 2 represents the diagram transitions for the node and the base station roles.

The properties we have to analyse are the following:

- Authentication of $Nonce$, Msg_1 and Msg_2 , i.e., the node N_i and the base station BS share the same value for $Nonce$, Msg_1 and Msg_2 and both execute the same session of the protocol. This property allows us to proof that bilateral authentication is achieved by using the MAC, and the integrity of the message is guarantied.
- Confidentiality of Msg_2, i.e.,Msg_2 is a secret value shared between N_i and BS, and they are not known by an intruder or third parties.

As the first message is not secure, there is the following attack:

1. I$_{BS}$ → N$_i$: $BS.N_i.Nonce_{false}.Msg_{false}$
2. N$_i$ → BS : $\{Ctr\}_{K_{enc}} \oplus Msg_2.$
   $\{MAC(\{Ctr\}_{K_{enc}} \oplus Msg_2.Nonce.N_i.BS)\}_{K_{mac}}$

The intruder creates a false request for a node. This message does not include any authentication information. Thus, the node believes that the message has been sent by the base station. The node responses the intruder request. The intruder cannot learn the value of Msg_2 because it has not learned the value of K_m. Even if the intruder captures another node, the value of K_m for node N_i is still secret. But, the attack implies resource and bandwith consumptions.

This attack can be prevent computing a MAC over the first message:

1. BS → N$_i$: $BS.N_i.Nonce.Msg_1.$
   $\{MAC(BS.N_i.Nonce.Msg_1)\}_{K_{mac}}$
2. N$_i$ → BS : $\{C_i\}_{K_{enc}} \oplus Msg_2.$
   $\{MAC(\{C_i\}_{K_{enc}} \oplus Msg_2.Nonce.N_i.BS)\}_{K_{mac}}$

AVISPA does not report about any attack in this version of the protocol.

B. Case Node → Node

A common approach for key distribution in networks is using asymmetric key protocols in order to exchange a shared symmetric key. In sensor networks, this approach is not feasible due to its resource consumption. Consequently, the nodes can only afford symmetric key algorithms. SPINS propose a key distribution solution based on base station intervention. The A-B notation for this protocol is the following:

1. A → B : $N_A.A.B$
2. B → BS : $N_A.N_B.A.B\{MAC(N_A.N_B.A.B)\}_{K_{mac}}$
3. BS → A : $\{C_{bs-a}\}_{K_{enc}} \oplus Sk_{ab}.$
   $\{MAC(\{C_{bs-a}\}_{K_{enc}} \oplus Sk_{ab}.N_a.B)\}_{K_{mac}}$
4. BS → B : $\{C_{bs-b}\}_{K_{enc}} \oplus Sk_{ab}.$
   $\{MAC(\{C_{bs-b}\}_{K_{enc}} \oplus Sk_{ab}.N_b.A)\}_{K_{mac}}$
5. A → B : $Msg_1.N_c\{MAC(Msg_1.N_c)\}_{F(Sk_{ab},1)}$
6. B → A : $\{C_{a-b}\}_{F(Sk_{ab},0)} \oplus Msg_2.$
   $\{MAC(\{C_{a-b}\}_{F(Sk_{ab},0)} \oplus Sk_{ab}.Msg_2.N_c)\}_{F(Sk_{ab},1)}$

The node A would like to share a key with the node B. Initially, both nodes do not share any secret. But both nodes share a key with the base station. Consequently, the base station should participate. The base station computes the new shared key and distributes the key to the nodes. Most of the computation takes place in the base station, so the nodes save resources.

The fig. 3 represents our model. There are two transition diagrams; one represents the node role and the other one for the base station. A node can plays the part of node A or node

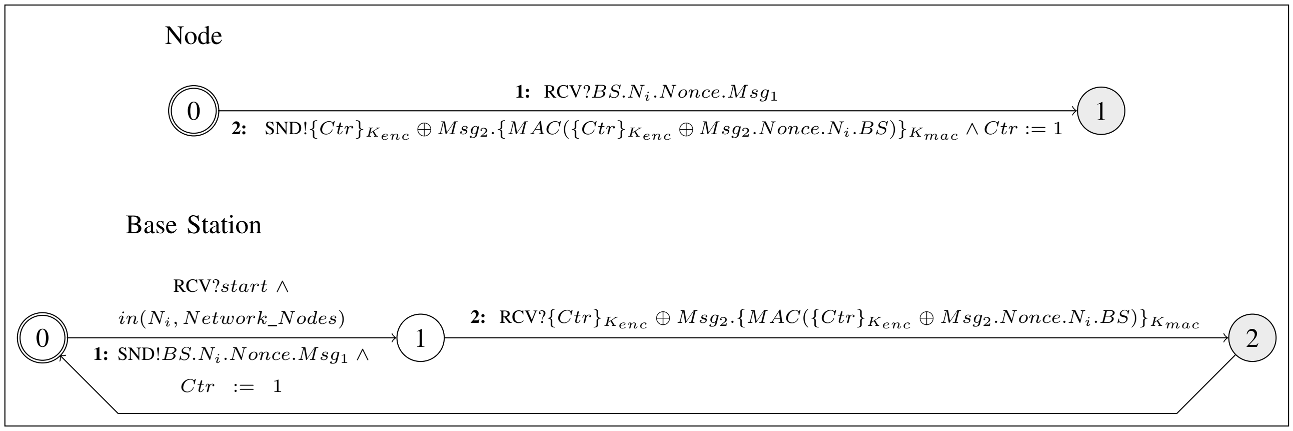
Node
1: RCV?$BS.N_i.Nonce.Msg_1$
2: SND!$\{Ctr\}_{K_{enc}} \oplus Msg_2.\{MAC(\{Ctr\}_{K_{enc}} \oplus Msg_2.Nonce.N_i.BS)\}_{K_{mac}} \wedge Ctr := 1$
Base Station
RCV?$start \wedge in(N_i, Network\_Nodes)$
1: SND!$BS.N_i.Nonce.Msg_1 \wedge Ctr := 1$
2: RCV?$\{Ctr\}_{K_{enc}} \oplus Msg_2.\{MAC(\{Ctr\}_{K_{enc}} \oplus Msg_2.Nonce.N_i.BS)\}_{K_{mac}}$
0
1
2

Figure 2. Model diagram

B, but once it choose one it can not perform the other part until the protocol is finished.

The properties we have to analyse are the following:

- Authentication of N_A and N_B , i.e., the node A, the node B and the base station BS share the same value for N_A and N_B and all of them execute the same session of the protocol. This property allows us to proof that there are not replays attacks and the freshness of messages is guarantee.
- Authentication of SK_{ab}, i.e., the node A, the node B and the base station BS share the same value for N_A, N_B and SK_{ab} and all of them execute the same session of the protocol. This property allows us to proof that node A and node B completes a bilateral authentication through the base station BS because they share the same key SK_{ab}.
- Authentication of Msg_1 and Msg_2, i.e., the node A and the node B share the same value for Msg_1 and Msg_2 and both execute the same session of the protocol. This property allows us to proof that bilateral authentication is achieved by using the MAC, and the integrity of the message is guarantied.
- Confidentiality of Msg_2, i.e.,Msg_2 is a secret value shared between A and B, and they are not known by an intruder or third parties.
- Confidentiality SK_{ab}, i.e., SK_{ab} is a secret value shared among A, B and BS, and they are not known by an intruder or third parties.

In this case AVISPA does not report about any attack.

V. Conclusion

In this paper we have presented a formal approach to the security analysis of wireless sensor networks by means of a model checking tool called Avispa.

The first case analyses the communication between the base station and the node. We have found an attack which leads resource and bandwith consumption. We have propose a solution to this attack. We can conclude that this amended version of the protocol guarantees the strong freshness, authentication, message integrity and confidentiality of the messages under our assumptions.

The second case is a key distribution protocol between two nodes. This protocol allows two nodes to establish a shared symmetric key through the base station. We have not found any attack on the protocol. So we can conclude is a secure solution for key distribution under our assumptions.

Our future work is concerned with extending our analysis to other security protocols for wireless sensor networks such as TinySec[2], μTESLA (defined in [3]), and MiniSec [20]. Also we are interested in the analysis of TinySec with other distribution key protocols such as LEAP+[21] and TinyPK [22].

References

[1] A. Perrig, J. A. Stankovic, and D. Wagner, "Security in wireless sensor networks." *Commun. ACM*, vol. 47, no. 6, pp. 53–57, 2004.
[2] C. Karlof, N. Sastry, and D. Wagner, "TinySec: a link layer security architecture for wireless sensor networks." in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, SenSys 2004, Baltimore, MD, USA, November 3-5, 2004*. ACM, 2004, pp. 162–175.
[3] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: Security protocols for sensor networks." *Wireless Networks*, vol. 8, no. 5, pp. 521–534, 2002.
[4] S. Zhu, S. Setia, and S. Jajodia, "LEAP: efficient security mechanisms for large-scale distributed sensor networks." in *ACM Conference on Computer and Communications Security*, S. Jajodia, V. Atluri, and T. Jaeger, Eds. ACM, 2003, pp. 62–72.
[5] H. Chan, A. Perrig, and D. X. Song, "Random key predistribution schemes for sensor networks." in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2003, p. 197.
[6] L. Eschenauer and V. Gligor, "A key-management scheme for distributed sensor networks." in *ACM Conference on Computer and Communications Security*, V. Atluri, Ed. ACM, 2002, pp. 41–47.
[7] A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar, "SPINS: security protocols for sensor netowrks," in *Mobile Computing and Networking*.
[8] E. M. Clarke, O. Grumberg, and D. A. Peled, *Model Checking*. The MIT Press, 1999.
[9] G. Lowe, "Casper: A compiler for the analysis of security protocols." *Journal of Computer Security*, vol. 6, no. 1-2, pp. 53–84, 1998.
[10] A. Armando, D. A. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuéllar, P. H. Drielsma, P.-C. Héam, O. Kouchnarenko, J. Mantovani, S. Mödersheim, D. von Oheimb, M. Rusinowitch, J. Santiago, M. Turuani, L. Viganò, and L. Vigneron, "The AVISPA tool for the automated validation of internet security protocols and applications." in *CAV*, ser.

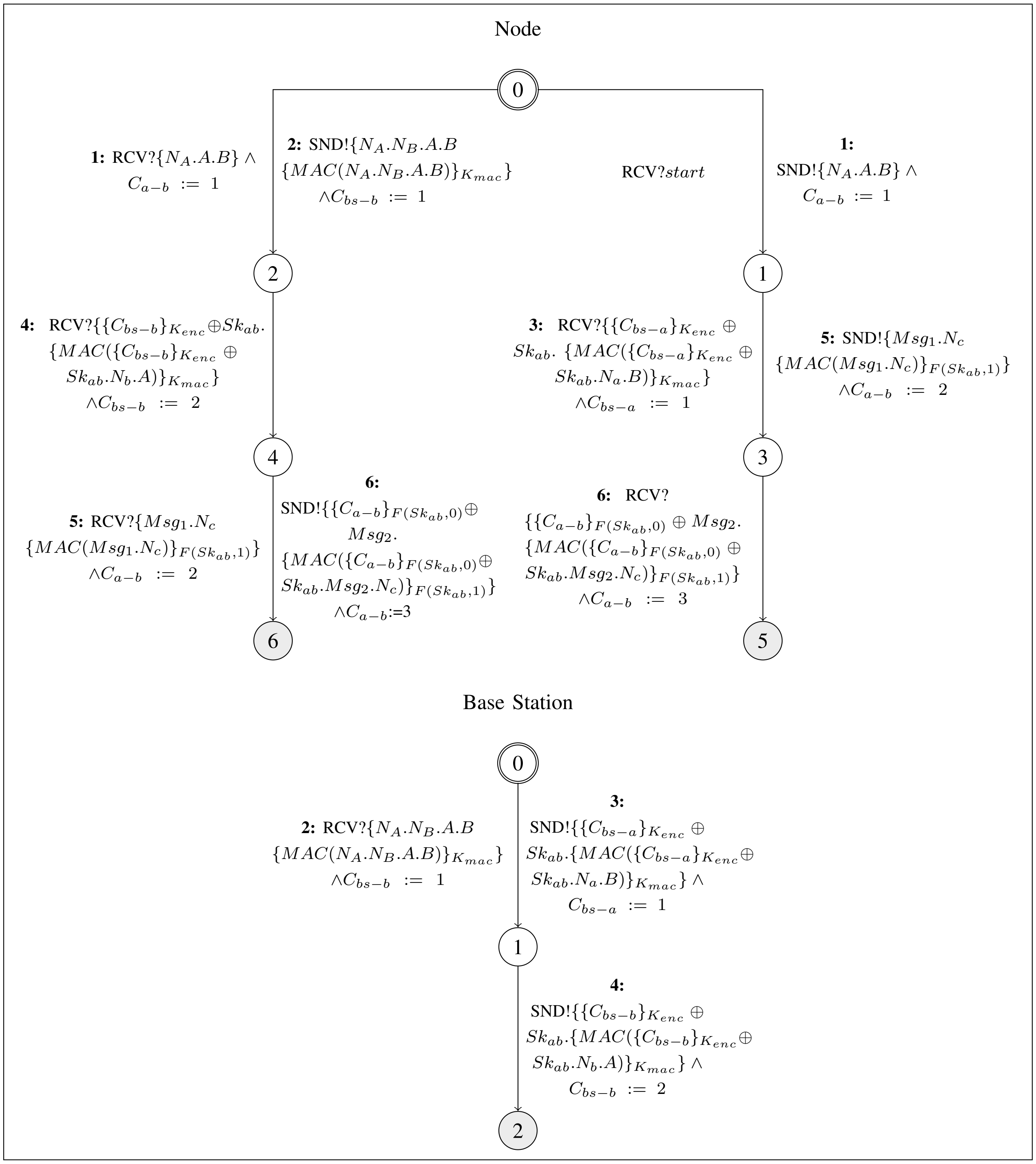
Node
0
1: RCV?$\{N_A.A.B\} \wedge C_{a-b} := 1$
2: SND!$\{N_A.N_B.A.B \{MAC(N_A.N_B.A.B)\}_{K_{mac}}\} \wedge C_{bs-b} := 1$
RCV?$start$
1: SND!$\{N_A.A.B\} \wedge C_{a-b} := 1$
2
1
4: RCV?$\{\{C_{bs-b}\}_{K_{enc}} \oplus Sk_{ab}. \{MAC(\{C_{bs-b}\}_{K_{enc}} \oplus Sk_{ab}.N_b.A)\}_{K_{mac}}\} \wedge C_{bs-b} := 2$
3: RCV?$\{\{C_{bs-a}\}_{K_{enc}} \oplus Sk_{ab}. \{MAC(\{C_{bs-a}\}_{K_{enc}} \oplus Sk_{ab}.N_a.B)\}_{K_{mac}}\} \wedge C_{bs-a} := 1$
5: SND!$\{Msg_1.N_c \{MAC(Msg_1.N_c)\}_{F(Sk_{ab},1)}\} \wedge C_{a-b} := 2$
4
3
5: RCV?$\{Msg_1.N_c \{MAC(Msg_1.N_c)\}_{F(Sk_{ab},1)}\} \wedge C_{a-b} := 2$
6: SND!$\{\{C_{a-b}\}_{F(Sk_{ab},0)} \oplus Msg_2. \{MAC(\{C_{a-b}\}_{F(Sk_{ab},0)} \oplus Sk_{ab}.Msg_2.N_c)\}_{F(Sk_{ab},1)}\} \wedge C_{a-b}$:=3
6: RCV?$\{\{C_{a-b}\}_{F(Sk_{ab},0)} \oplus Msg_2. \{MAC(\{C_{a-b}\}_{F(Sk_{ab},0)} \oplus Sk_{ab}.Msg_2.N_c)\}_{F(Sk_{ab},1)}\} \wedge C_{a-b} := 3$
6
5
Base Station
0
2: RCV?$\{N_A.N_B.A.B \{MAC(N_A.N_B.A.B)\}_{K_{mac}}\} \wedge C_{bs-b} := 1$
3: SND!$\{\{C_{bs-a}\}_{K_{enc}} \oplus Sk_{ab}.\{MAC(\{C_{bs-a}\}_{K_{enc}} \oplus Sk_{ab}.N_a.B)\}_{K_{mac}}\} \wedge C_{bs-a} := 1$
1
4: SND!$\{\{C_{bs-b}\}_{K_{enc}} \oplus Sk_{ab}.\{MAC(\{C_{bs-b}\}_{K_{enc}} \oplus Sk_{ab}.N_b.A)\}_{K_{mac}}\} \wedge C_{bs-b} := 2$
2

Figure 3. Model diagram

Lecture Notes in Computer Science, K. Etessami and S. K. Rajamani, Eds., vol. 3576. Springer, 2005, pp. 281–285.

[11] M. L. Tobarra, D. Cazorla, F. Cuartero, and G. Diaz, “Application of formal methods to the analysis of web services security.” in *EPEW/WS-FM*, ser. Lecture Notes in Computer Science, M. Bravetti, L. Kloul, and G. Zavattaro, Eds., vol. 3670. Springer, 2005, pp. 215–229.

[12] M. Backes, S. Mödersheim, B. Pfitzmann, and L. Viganò, “Symbolic and cryptographic analysis of the secure WS-ReliableMessaging scenario.” in *FoSSaCS*, ser. Lecture Notes in Computer Science, L. Aceto and A. Ingólfsdóttir, Eds., vol. 3921. Springer, 2006, pp. 428–445.

[13] K. Bhargavan, C. Fournet, and A. D. Gordon, “Verifying policy-based security for web services.” in *ACM Conference on Computer and Communications Security*, V. Atluri, B. Pfitzmann, and P. D. McDaniel, Eds. ACM, 2004, pp. 268–277.

[14] I.-G. Kim and J.-Y. Choi, “Formal verification of pap and eap-md5 protocols in wireless networks: Fdr model checking,” *aina*, vol. 02, p. 264, 2004.

[15] J. C. Mitchell, “Finite-state analysis of security protocols.” in *CAV*, ser. Lecture Notes in Computer Science, A. J. Hu and M. Y. Vardi, Eds., vol. 1427. Springer, 1998, pp. 71–76.

[16] M. L. Tobarra, D. Cazorla, F. Cuartero, and G. Diaz, “Formal verification of TLS handshake and extensions for wireless networks,” in *Proc. of IADIS International Conference on Applied Computing (AC’06)*. San Sebastian, Spain: IADIS Press, February 2006, pp. 57–64.

[17] Y. Chevalier, L. Compagna, J. Cuéllar, P. H. Drielsma, J. Mantovani, S. Mödersheim, and L. Vigneron, “A high level protocol specification language for industrial security-sensitive protocols.” in *Proceedings of Workshop on Specification and Automated Processing of Security Requirements (SAPS)*, 2004, pp. 193–205.

[18] Y. Glouche, T. Genet, O. Heen, and O. Courtay, “A security protocol animator tool for AVISPA,” in *ARTIST2 Workshop on Security Specification and Verification of Embedded Systems*, Pisa, May 2006.

[19] D. Dolev and A. C.-C. Yao, “On the security of public key protocols,” in *FOCS*. IEEE, 1981, pp. 350–357.

[20] A. P. Mark Luk, Ghita Mezzour and V. Gligor, “Minisec: A secure sensor network communication architecture,” in *Proceedings of IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, April 2007. [Online]. Available: http://www.truststc.org/pubs/197.html

[21] S. Zhu, S. Setia, and S. Jajodia, “LEAP : Efficient security mechanisms for large-scale distributed sensor networks.” *ACM Transactions on Sensor Networks*, vol. 2, no. 4, pp. 500–528, 2006.

[22] R. J. Watro, D. Kong, S.-f. Cuti, C. Gardiner, C. Lynn, and P. Kruus, “TinyPK: securing sensor networks with public key technology.” in *SASN*, S. Setia and V. Swarup, Eds. ACM, 2004, pp. 59–64.